

1

Introdução à Programação

ESQUEMA DO CAPÍTULO

1.1 EXERCÍCIO DE PROGRAMAÇÃO

1.2 SISTEMAS DE PROGRAMAÇÃO

1.3 ALGORITMOS E FLUXOGRAMAS

1.4 ESTRUTURAS DE DADOS

1.5 MODULARIZAÇÃO

1.3 Algoritmos e Fluxogramas

1.3.1. conceito

Definição:

Algoritmo é a descrição de um conjunto de comandos que, obedecidos, resultam numa sucessão finita de ações. Geralmente, um algoritmo se destina a resolver um problema (Farrer et al., 1999).

PROGRAMAR = CONSTRUIR ALGORITMOS

a) versão inicial

descasque as batatas para o jantar

b) sequência de sub-ações:

traga a cesta com batatas

traga a panela

descasque as batatas

c) repetição:

traga a cesta com batatas

traga a panela

enquanto número de batatas é insuficiente faça

descasque a batata

fim enquanto

1.3 Algoritmos e Fluxogramas

1.3.1. conceito

d) condicional

traga a cesta com batatas

traga a panela

enquanto número de batatas é insuficiente faça

se batata está boa então

 descasque esta batata

senão

 descarte a batata

fim se

fim enquanto

e) algoritmo impróprio

enquanto batata está boa faça

 descasque esta batata

fim enquanto

1.3 Algoritmos e Fluxogramas

1.3.1. conceito

MÁQUINA = SEGUIDOR MECÂNICO DE ALGORITMOS

PROGRAMA = ALGORITMO DESTINADO A CONTROLAR O COMPORTAMENTO DA MÁQUINA

algoritmo:

- ações + dados

1.3 Algoritmos e Fluxogramas

1.3.2. estruturas básicas de dados e controle (IMPORTANT!)

variável:

posição na memória que tem nome e valor associados:

NOTA

5

declaração de variáveis:

- declare *lista de variáveis* tipo

tipos de dados:

- numérico
- literal
- lógico

exemplo:

declare x logico;
vendas, salario numérico;

1.3 Algoritmos e Fluxogramas

1.3.2. estruturas básicas de dados e controle (IMPORTANT!)

estruturas de controle:

1. sequência de comandos:

```
comando 1  
comando 2
```

2. comando condicional:

```
se condição então  
    sequência de comandos 1  
senão  
    sequência de comandos 2  
fim se
```

1.3 Algoritmos e Fluxogramas

1.3.2. estruturas básicas de dados e controle (IMPORTANT!)

estruturas de controle (cont):

3. comandos de repetição:

3.1 com interrupção no início (“enquanto”)

```
enquanto condição faça  
    sequência de comandos  
fim enquanto
```

3.2 comando de repetição com interrupção no final (“repita”)

```
repita  
    sequência de comandos  
ate condição
```

3.3 comando de repetição com número fixo de vezes (“para”)

```
para variável de valor1 até valor2 faça  
    sequência de comandos  
fim para
```

1.3 Algoritmos e Fluxogramas

1.3.2. estruturas básicas de dados e controle (IMPORTANT!)

estruturas de controle (final):

4. comando de entrada e saída:

leia *lista de variáveis*
escreva *lista de variáveis*

5. atribuição:

$K \leftarrow 10$

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Ex. 1:

Fazer um algoritmo que leia três valores inteiros, determine e imprima o menor deles.

Solução:

Um algoritmo possível é apresentado abaixo:

```
algoritmo
    defina os tipos das variáveis
    leia os números
    determine o menor número
    escreva o menor número
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Refinamentos:

```
ref.: leia os números  
        leia a, b, c  
fim ref.
```

```
ref.: determine o menor número  
        se  $a < b$  e  $a < c$  então  
            menor  $\leftarrow$  a  
        senão  
            determine o menor entre b e c  
        fim se  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Refinamentos:

ref.: determine o menor entre b e c

se $b < c$ então

menor $\leftarrow$ b

senão

menor $\leftarrow$ c

fim se

fim ref.

ref.: escreva o menor número

escreva menor

fim ref.

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Refinamentos:

```
ref.: defina os tipos das variáveis  
      declare a, b, c, menor numerico  
fim ref.
```

Agora, inserindo-se os refinamentos nos seus respectivos lugares e colocando-se comentários, tem-se:

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

```
algoritmo
    {defina os tipos das variáveis}
    declare a, b, c, menor numérico
    {leia os números}
    leia a, b, c
    {determine o menor número}
    se a<b e a<c então
        menor ← a
    senão
        se b<c então
            menor ← b
        senão
            menor ← c
    fim se
    fim se
    {escreva o menor número}
    escreva menor
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Ex. 2:

Fazer um algoritmo que calcule o valor de $n!$ (fatorial de n) sendo que o valor inteiro de n se encontra disponível em uma unidade de entrada e que:

- a) $n! = 1 \times 2 \times \dots \times (n-1) \times n$; e
- b) $0! = 1$, por definição.

Solução:

Um algoritmo possível é descrito abaixo:

```
algoritmo  
  defina os tipos das variáveis  
  leia n  
  calcule o fatorial de n  
  escreva fatorial  
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Os refinamentos são apresentados abaixo:

```
ref.: calcule fatorial de n  
    fatorial  $\leftarrow$  1  
    i  $\leftarrow$  2  
    enquanto i  $\leq$  n faca  
        fatorial  $\leftarrow$  fatorial * i  
        i  $\leftarrow$  i + 1  
    fim enquanto  
fim ref.
```

```
ref.: defina os tipos das variáveis  
    declare i, n, fatorial numérico  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Agora, inserindo-se o refinamento anterior no seu respectivo lugar, tem-se o algoritmo completo:

```
algoritmo  
  declare i, n, fatorial numérico  
  leia n  
  fatorial  $\leftarrow$  1  
  i  $\leftarrow$  2  
  enquanto i  $\leq$  n faça  
    fatorial  $\leftarrow$  fatorial * i  
    i  $\leftarrow$  i + 1  
  fim enquanto  
  escreva fatorial  
fim algoritmo
```


1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Ex. 3:

Fazer um algoritmo que:

- a) leia de um arquivo e escreva o nome e a altura das moças inscritas em uma seleção de ponteira para um time de voleibol;
- b) para cada moça, existe uma linha contendo o seu nome e a sua altura; a última linha, que não corresponde a nenhuma moça, conterá a palavra “vazio” no lugar do nome;
- c) calcule e escreva as duas maiores alturas e quantas moças as possuem.

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução:

A seguir, é apresentado um algoritmo possível:

```
algoritmo  
  defina os tipos das variáveis  
  atribua valores iniciais necessários  
  leia nome, altura  
  enquanto nome <> "vazio" faca  
    escreva nome, altura  
    compare altura com as duas maiores alturas  
    leia nome, altura  
  fim enquanto  
  escreva resultados  
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

E os refinamentos sucessivos são:

```
ref.: compare altura com as duas maiores alturas
      se altura > maior1 então
        atualize a 2ª maior altura e o número de moças que a possui
        atualize a 1ª maior altura e o número de moças que a possui
      senão
        verifique se altura é igual a maior1
      fim se
fim ref.
```

```
ref.: atualize a 2ª maior altura e o número de moças que a possui
      maior2 ← maior1
      contador2 ← contador1
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

ref.: atualize a 1ª maior altura e o número de moças que a possui
maior1 $\leftarrow$ altura
contador1 $\leftarrow$ 1
fim ref.

ref.: verifique se altura é igual a maior1
se altura = maior1 então
contador1 $\leftarrow$ contador1 + 1
senão
compare altura com a 2ª maior altura
fim se
fim ref.

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

```
ref.: compare altura com a 2ª maior altura  
  se altura > maior2 então  
    maior2 ← altura  
    contador2 ← 1  
  senão  
    se altura = maior2 então  
      contador2 ← contador2 + 1  
    fim se  
  fim se  
fim ref.
```

```
ref.: escreva resultados  
  escreva maior1, contador1, maior2, contador2  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

```
ref.: atribua valores iniciais necessários  
    contador1 ← 0  
    maior1 ← 0  
    contador2 ← 0  
    maior2 ← 0  
fim ref.
```

```
ref.: defina tipo das variáveis  
    declare altura, contador1, contador2, maior1, maior2 numérico  
    declare nome literal  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.3. desenvolvimento de algoritmos por refinamentos sucessivos

Solução (cont.):

Inserindo-se os refinamentos nos devidos lugares, teremos:

```
algoritmo
  declare altura, contador1, contador2, maior1, maior2 numérico
  declare nome literal
  contador1 ← 0
  maior1 ← 0
  contador2 ← 0
  maior2 ← 0
  leia nome, altura
  enquanto nome <> "vazio" faca
    escreva nome, altura
    se altura > maior1 então
      maior2 ← maior1
      contador2 ← contador1
      maior1 ← altura
      contador1 ← 1
    senão
      se altura = maior1 então
        contador1 ← contador1 + 1
      senão
        se altura > maior2 então
          maior2 ← altura
          contador2 ← 1
        senão
          se altura = maior2 então
            contador2 ← contador2 + 1
          fim se
        fim se
      fim se
    leia nome, altura
  fim enquanto
  escreva maior1, contador1, maior2, contador2
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.4. exercício

Desenvolver um algoritmo para ler diversos coeficientes a , b e c , calcular e escrever as raízes da equação $ax^2 + bx + c = 0$. Parar quando encontrar um coeficiente $a = 0$.

1.3 Algoritmos e Fluxogramas

1.3.4. exercício – solução:

Uma solução possível para o problema é apresentada a seguir.

Inicialmente, desenvolvemos uma primeira versão do algoritmo em que já identificamos a estrutura de repetição que deverá estar presente. Note que deixamos para mais tarde alguns detalhes (isto é, deixamos para os refinamentos):

```
algoritmo
  defina os tipos das variáveis
  leia a, b, c
  enquanto a<>0 faça
    calcule raízes
    escreva raízes
    leia a, b, c
  fim enquanto
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.4. exercício – solução (cont.):

Refinando-se o cálculo das raízes, temos:

```
ref.: calcule raízes  
    delta  $\leftarrow b*b - 4*a*c$   
    se delta  $\geq 0$  então  
        xr1  $\leftarrow (-b + \text{raiz}(\text{delta})) / (2*a)$   
        xr2  $\leftarrow (-b - \text{raiz}(\text{delta})) / (2*a)$   
        xi1  $\leftarrow 0$   
        xi2  $\leftarrow 0$   
    senão  
        xr1  $\leftarrow -b / (2*a)$   
        xr2  $\leftarrow -b / (2*a)$   
        xi1  $\leftarrow \text{raiz}(-\text{delta}) / (2*a)$   
        xi2  $\leftarrow -\text{raiz}(-\text{delta}) / (2*a)$   
    fim se  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.4. exercício – solução (cont.):

Refinando-se a escrita das raízes:

```
ref.: escreva raízes  
    escreva “Raiz 1:”, xr1, “+”, xi1, “i”  
    escreva “Raiz 2:”, xr2, “-”, xi2, “i”  
fim ref.
```

Finalmente, refinando-se a declaração das variáveis:

```
ref.: defina os tipos da variáveis  
    declare a, b, c, xr1, xi1, xr2, xi2, delta numérico  
fim ref.
```

1.3 Algoritmos e Fluxogramas

1.3.4. exercício – solução (cont.):

Inserindo-se os refinamentos nos respectivos lugares, temos o algoritmo completo:

```
algoritmo
    {defina os tipos das variáveis}
    declare a, b, c, xr1, xi1, xr2, xi2, delta numérico
    leia a, b, c
    enquanto a <> 0 faça
        {calcule raízes}
        delta  $\leftarrow b*b - 4*a*c$ 
        se delta >= 0 então
            xr1  $\leftarrow (-b + \text{raiz}(\text{delta})) / (2*a)$ 
            xr2  $\leftarrow (-b - \text{raiz}(\text{delta})) / (2*a)$ 
            xi1  $\leftarrow 0$ 
            xi2  $\leftarrow 0$ 
        senão
            xr1  $\leftarrow -b / (2*a)$ 
            xr2  $\leftarrow -b / (2*a)$ 
            xi1  $\leftarrow \text{raiz}(-\text{delta}) / (2*a)$ 
            xi2  $\leftarrow -\text{raiz}(-\text{delta}) / (2*a)$ 
        fim se
        {escreva raízes}
        escreva "Raiz 1:", xr1, "+", xi1, "i"
        escreva "Raiz 2:", xr2, "-", xi2, "i"
        leia a, b, c
    fim enquanto
fim algoritmo
```

1.3 Algoritmos e Fluxogramas

1.3.4. exercício – solução (cont.):

Na linguagem do R, o algoritmo pode ser codificado conforme apresentado a seguir, ou em <http://www.est.ufmg.br/ftp/fcruz/iec/eq2grau.R>:

```
eq2grau <- function() {  
  repeat {  
    cat("Entre com o coef a ")  
    a <- as.numeric(readline())  
    cat("Entre com o coef b ")  
    b <- as.numeric(readline())  
    cat("Entre com o coef c ")  
    c <- as.numeric(readline())  
    if (a==0) {  
      break  
    }  
    delta <- b^2 - 4*a*c  
    if (delta >= 0) {  
      xr1 <- (-b + sqrt(delta))/(2*a)  
      xr2 <- (-b - sqrt(delta))/(2*a)  
      xi1 <- 0  
      xi2 <- 0  
    } else {  
      xr1 <- -b/2*a  
      xr2 <- -b/2*a  
      xi1 <- sqrt(-delta)/(2*a)  
      xi2 <- -sqrt(-delta)/(2*a)  
    }  
    cat("x1 = (", xr1, "+", xi1, "i)\n")  
    cat("x2 = (", xr2, "-", xi2, "i)\n")  
  }  
}
```